Short binary convolutional codes with maximal free distance for rates 2/3 and 3/4

Paaske, Erik

Published in:
I E E E Transactions on Information Theory

Publication date:
1974

Document Version
Publisher's PDF, also known as Version of record

Lemma 1: Let C be an (n,k) cyclic q^m-ary code, let ψ be a linear mapping from the vector space $GF(q^m)$ over $GF(q)$ onto the vector space $GF(q^r)$ over $GF(q)$, and let $\psi(C) = \{(\psi(u_0), \psi(u_1),\cdots,\psi(u_{n-1})) \mid (u_0,u_1,\cdots,u_{n-1}) \in C\}$. Assume that $\psi(C)$ is a linear code over $GF(q^r)$ with $\bar{k}$ information digits. Then $\bar{k} \geq k$, where the equality holds only if the check polynomial of C has coefficients in $GF(q^{(m,r)})$.

Proof: As is known [9], ψ can be represented as follows:

$$\psi(\omega) = \sum_{i=1}^{r} \delta_i T_{q,m}(\lambda_i \omega)$$

where $\omega \in GF(q^m)$, $\lambda_i \in GF(q^m)$, for $1 \leq i \leq r$, and $\{\delta_1,\cdots,\delta_r\}$ is a basis of the vector space $GF(q^r)$ over $GF(q)$. There is $\omega_0 \in GF(q^m)$ such that $T_{q,m}(\lambda_1\omega_0) \neq 0$. Let ψ_0 denote the linear mapping from $GF(q^m)$ onto $GF(q^m)$ defined by

$$\psi_0(\omega) = \omega_0^{-1}\omega \qquad \omega \in GF(q^m).$$

Since $\psi_0(C) = C$, we can consider $\psi\psi_0^{-1}$ instead of ψ and there is no loss of generality in assuming that $T_{q,m}(\lambda_1) \neq 0$. Since $T_{q,m}(\lambda_i) \in GF(q)$ and $\{\delta_1,\delta_2,\cdots,\delta_r\}$ is a basis

$$\sum_{i=1} \delta_i T_{q,m}(\lambda_i) \neq 0. \tag{21}$$

By the assumption, $\psi(C)$ is cyclic. Let $h(x)$ and $\bar{h}(x)$ denote the check polynomials of C and $\psi(C)$, respectively, and let $J = \{j \mid h(\gamma^j) = 0,\ 0 \leq j < n\}$ and $\bar{J} = \{j \mid \bar{h}(\gamma^j) = 0,\ 0 \leq j < n\}$, where γ is a primitive nth root of unity. The Mattson–Solomon polynomial of a codeword of C is of the following form:

$$\phi(Z) = \sum_{j=0}^{n-1} a_j Z^{n-j}$$

where $a_j = 0$, for $j \notin J$, $a_{jq^m} = a_j^{q^m}$, and $a_j \in GF(q^{mM(q^m,n/(n,j))})$, for $j \in J$. Since $(\psi(\phi(\gamma^0)),\psi(\phi(\gamma^1)),\cdots,\psi(\phi(\gamma^{n-1})) \in \psi(C)$ and the Mattson–Solomon polynomial of a codeword is unique,[2] $\psi(\phi(Z))$ is the Mattson–Solomon polynomial of a codeword of C. Since the exponent of Z is to be taken modulo n, we have that

$$\begin{aligned}
\psi(\phi(Z)) &= \sum_{i=1}^{r} \delta_i T_{q,m}\left(\lambda_i \sum_{j=0}^{n-1} a_j Z^{n-j}\right) \\
&= \sum_{i=1}^{r} \delta_i \sum_{s=0}^{m-1} \left(\lambda_i \sum_{j=0}^{n-1} a_j Z^{n-j}\right)^{q^s} \\
&= \sum_{i=1}^{r} \delta_i \sum_{j=0}^{n-1} \left(\sum_{s=0}^{m-1} \lambda_i^{q^s} a_{jq^s}^{q^s}\right) Z^{n-j} \\
&= \sum_{j=0}^{n-1} \left[\sum_{s=0}^{m-1} a_{jq^s}^{q^s} \left(\sum_{i=1}^{r} \delta_i \lambda_i^{q^s}\right)\right] Z^{n-j}
\end{aligned}$$

where the suffix of a is taken modulo n. Now, we will express the coefficient b_j of Z^{n-j} of $\psi(\phi(Z))$ as a polynomial of those coefficients of $\phi(Z)$, which can be chosen independently. Let $M_j = M(q,n/(n,j))$. Then $jq^{-s} \equiv jq^{mt} \bmod n$, if and only if $M_j \mid mt + s$. Hence $jq^{-s} \equiv jq^{mt} \bmod n$ for some integer t, if and only if $\mid m_j s$, where $m_j = (m,M_j)$. For $0 \leq s < m_j$, let

$$c_{js} = \sum_{t=0}^{m/m_j-1} \sum_{i=1}^{r} \delta_i \lambda_i^{qs+tm_j}.$$

Since

$$\sum_{s=0}^{m_j-1} c_{js} = \sum_{i=1}^{r} \delta_i T_{q,m}(\lambda_i) \neq 0$$

by (21), there is an s_j such that

$$c_{js_j} \neq 0, \qquad 0 \leq s_j < m_j.$$

For $0 \leq j < n$ such that $(jq^{-s_j})_n \in J$, take $T_{q^m,M(q^m,n/(n,j))}(Z^{n-j'})$ as $\phi(Z)$, where $j' = (jq^{-s_j})_n$. Then $a_{jq^{-s_j}-m_jt} = 1$, for $0 \leq t < m/m_j$, and $a_l = 0$ for other i. Hence

$$b_j = \sum_{t=0}^{m/m_j-1} \sum_{i=1}^{r} \delta_i \lambda_i^{qs_j+m_jt} = c_{js_j} \neq 0$$

and therefore

$$j \in \bar{J}. \tag{22}$$

For an integer i relatively prime to n, let π_i denote the permutation $j \to ij \pmod n$ of the set of integers $\{0, 1,\cdots, n-1\}$, let $N_1,N_2,\cdots,N_\rho$ denote the cycles of the permutation π_q, and let $J_t = J \cap N_t$ and $\bar{J}_t = \bar{J} \cap N_t$, for $1 \leq t \leq \rho$. For j and j' in N_t, $m_j = m_{j'}$, and $s_j = s_{j'}$. Let $s(t) = s_j$, for $j \in N_t$. Then, it follows from (22) that

$$\pi_{qs(t)} J_t \subseteq \bar{J}_t, \qquad \text{for } 1 \leq t \leq \rho$$

where $\pi_{qs(t)}J_t = \{(q^{s(t)}j)_n \mid j \in J_t\}$. Hence $k \leq \bar{k}$. Suppose that $k = \bar{k}$. Then $\pi_{qs(t)}J_t = \bar{J}_t$, and $J_t = \pi_{q-s(t)}\bar{J}_t$, for $1 \leq t \leq \rho$. Since J_t is closed under π_{q^m}, and $\bar{J}_t$ is closed under π_{q^r}, J_t must be closed under $\pi_{q^{(m,r)}}$.

Acknowledgment

The author wishes to thank Prof. E. R. Berlekamp for a copy of a preliminary version of his paper [1] and S. Azumi for his help in preparing this paper.

References

[1] E. R. Berlekamp and J. Justesen, "Some long cyclic binary codes are not so bad," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 351–356, May 1974.
[2] J. M. Goethals, "Factorization of cyclic codes," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 242–246, Apr. 1967.
[3] T. Kasami, "Some lower bounds on the minimum weight of cyclic codes of composite length," *IEEE Trans. Inform. Theory*, vol. IT-14, pp. 814–818, Nov. 1968.
[4] H. F. Mattson and G. Solomon, "A new treatment of Bose–Chaudhuri codes," *J. Soc. Ind. Appl. Math.*, vol. 9, no. 4, pp. 654–669, 1961.
[5] E. R. Berlekamp, *Algebraic Coding Theory*. New York: McGraw-Hill, 1968.
[6] W. W. Peterson and E. J. Weldon, Jr., *Error Correcting Codes*, 2nd ed. Cambridge, Mass.: M.I.T. Press, 1972.
[7] G. D. Forney, Jr., *Concatenated Codes*. Cambridge, Mass.: M.I.T. Press, 1966.
[8] E. R. Berlekamp, "Long primitive binary BCH codes have distance $d \sim 2n \ln R^{-1}/\log n$," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 415–426, May 1972.
[9] P. Delsarte, J. M. Goethals, and F. J. MacWilliams, "On generalized Reed–Muller codes and their relatives," *Inform. Contr.*, vol. 16, pp. 403–442, July 1970.

[2] $\psi(\phi(Z))$ is taken modulo $Z^n - 1$.

Short Binary Convolutional Codes with Maximal Free Distance for Rates 2/3 and 3/4

ERIK PAASKE, MEMBER, IEEE

***Abstract*—A search procedure is developed to find good short binary $(N,N-1)$ convolutional codes. It uses simple rules to discard from the complete ensemble of codes a large fraction whose free distance d_{free} either cannot achieve the maximum value or is equal to d_{free} of some code in the remaining set. Further, the search among the remaining codes is started in a subset in which we expect the possibility of finding codes with large values of d_{free} to be good. A number of short, optimum (in the sense of maximizing d_{free}), rate-2/3 and 3/4 codes found by the search procedure are listed.**

Manuscript received July 9, 1973; revised December 22, 1973.

The author is with the Institute of Circuit Theory and Telecommunication, Technical University of Denmark, Lyngby, Denmark.

I. Introduction

In the past very little progress has been made in finding "good" convolutional codes for rates greater than 1/2. However, Forney [1] has recently developed a linear correspondence between the states of a rate-K/N convolutional encoder $\boldsymbol{G}$ and the states of a corresponding syndrome former $\boldsymbol{H}^T$, where $\boldsymbol{H}$ is an encoder of the code dual to the code generated by $\boldsymbol{G}$. With this correspondence in mind it is reasonable to assume that the task of finding optimal rate $(N-1)/N$ codes would be no more difficult than the task of finding optimal rate $1/N$ codes. The short, but optimum (in the sense of maximizing the free distance), rate-2/3 and 3/4 codes found and listed in this correspondence verify this assumption.

II. Notation and Definitions

Most of the notation and definitions follow Forney [2], [3] and/or Costello [4], but only binary codes are considered.

We shall represent the input and output sequences of an $(N, N-1)$ convolutional encoder [2] by

$$\begin{aligned}\boldsymbol{x} &= [\boldsymbol{x}_0,\boldsymbol{x}_1,\boldsymbol{x}_2\cdots] \\ &= [x_0{}^1x_0{}^2x_0{}^3\cdots x_0^{N-1},x_1{}^1x_1{}^2\cdots x_1^{N-1},\cdots] \\ \boldsymbol{y} &= [\boldsymbol{y}_0,\boldsymbol{y}_1,\boldsymbol{y}_2\cdots] \\ &= [y_0{}^1y_0{}^2y_0{}^3\cdots y_0{}^N,y_1{}^1y_1{}^2\cdots y_1{}^N,\cdots]\end{aligned}$$

respectively, where $\boldsymbol{x}_i$ is the $(N-1)$-tuple of input digits at time i, $x_i{}^j$ is the input digit at time i in the jth input sequence, and similarly for the output sequence.

The transform of the input sequence will be written as an $(N-1)$-tuple

$$\boldsymbol{x}(D) = [x^1(D),x^2(D),\cdots,x^{N-1}(D)]$$

where $x^j(D) = x_0{}^j + x_1{}^jD + x_2{}^jD^2 + \cdots$, $1 \le j \le N-1$ is the transform of the jth input sequence. The sequence $\boldsymbol{x}$ is said to be rational if each of these transforms is a rational function, i.e., a ratio of polynomials. The transform of the output sequence is written in a similar way.

The encoding equations for a convolutional encoder over the set of all causal rational input sequences can be written as

$$\boldsymbol{y} = \boldsymbol{x}\boldsymbol{G}$$

where the semi-infinite matrix

$$\boldsymbol{G} = \begin{pmatrix} \boldsymbol{G}_0 & \boldsymbol{G}_1 & \boldsymbol{G}_2 & \cdots & & \\ \boldsymbol{0} & \boldsymbol{G}_0 & \boldsymbol{G}_1 & \boldsymbol{G}_2 & \cdots & \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{G}_0 & \boldsymbol{G}_1 & \boldsymbol{G}_2 & \cdots \\ & & \ddots & \ddots & \ddots & \end{pmatrix}$$

is called the generator matrix, the ith row of $\boldsymbol{G}$ is called the ith generator $\boldsymbol{g}_i$, and

$$\boldsymbol{G}_l = \begin{pmatrix} g_{l1}{}^1 & g_{l1}{}^2 & \cdots & g_{l1}{}^N \\ g_{l2}{}^1 & g_{l2}{}^2 & \cdots & g_{l2}{}^N \\ \vdots & \vdots & & \vdots \\ g_{l,N-1}^1 & g_{l,N-1}^2 & \cdots & g_{l,N-1}^N \end{pmatrix}$$

is an $(N-1) \times N$ matrix of binary elements.

To represent the convolutional encoder we also use the $(N-1) \times N$ matrix $\boldsymbol{G}(D)$, where

$$\boldsymbol{G}(D) = \begin{pmatrix} G_1{}^1(D) & G_1{}^2(D) & \cdots & G_1{}^N(D) \\ G_2{}^1(D) & G_2{}^2(D) & \cdots & G_2{}^N(D) \\ \vdots & \vdots & & \vdots \\ G_{N-1}^1(D) & G_{N-1}^2(D) & \cdots & G_{N-1}^N(D) \end{pmatrix}$$

and

$$G_i{}^j(D) = g_{0i}{}^j + g_{1i}{}^jD + g_{2i}{}^jD^2 + \cdots$$

is the transform of the generator sequence $g_{0i}{}^j, g_{1i}{}^j, g_{2i}{}^j\cdots$ $1 \le i \le N-1$, $1 \le j \le N$. Each $G_i{}^j(D)$ is called a generator polynomial. Then the encoding equations can be written in D-operator form as

$$\boldsymbol{y}(D) = \boldsymbol{x}(D)\boldsymbol{G}(D)$$

with all operations to be performed over $GF(2)$.

We shall further make use of the parity matrix $\boldsymbol{H}$, which is the semi-infinite matrix

$$\boldsymbol{H} = \begin{pmatrix} \boldsymbol{H}_0 & \boldsymbol{0} & \boldsymbol{0} & \cdots \\ \boldsymbol{H}_1 & \boldsymbol{H}_0 & \boldsymbol{0} & \cdots \\ \boldsymbol{H}_2 & \boldsymbol{H}_1 & \boldsymbol{H}_0 & \cdots \\ \vdots & \vdots & \vdots & \end{pmatrix}$$

where

$$\boldsymbol{H}_l = [h_l{}^1h_l{}^2\cdots h_l{}^N].$$

Corresponding to $\boldsymbol{G}(D)$ we have $\boldsymbol{H}(D)$, an N-tuple which can be written

$$\boldsymbol{H}(D) = [H^1(D)H^2(D)\cdots H^N(D)]$$

where $H^j(D) = h_0{}^j + h_1{}^jD + h_2{}^jD^2 + \cdots + h_v{}^jD^v$ is the transform of the parity-check sequence $h_0{}^j, h_1{}^j, h_2{}^j,\cdots,h_v{}^j$, $1 \le j \le N$.

The parity-check equations of the convolutional code can be written either as

$$\boldsymbol{G}\boldsymbol{H}^T = \boldsymbol{0}$$

where $\boldsymbol{0}$ is the semi-infinite all-zero matrix, or as

$$\boldsymbol{G}(D)\boldsymbol{H}^T(D) = \boldsymbol{0}(D)$$

where $\boldsymbol{0}(D)$ is the matrix of all-zero polynomials.

The next two definitions are due to Forney [2], but we state them explicitly because of their importance in the sequel.

Definition 1: The constraint length for input i is

$$v_i = \max_{1 \le j \le N} \{\deg [G_i{}^j(D)]\}$$

and the overall constraint length is

$$v = \sum_{i=1}^{N-1} v_i.$$

Definition 2: A basic encoder $\boldsymbol{G}(D)$ is *minimal* if its overall constraint length v is the obvious realization is equal to the maximum degree μ of its $(N-1) \times (N-1)$ subdeterminants.

A basic encoder [2] requires v memory elements in the obvious realization, but if $\mu < v$ there is another realization with μ (but no fewer than μ) memory elements. For a minimal encoder the obvious realization requires only the minimum number of memory elements, and since Forney [2] has shown that any encoder is equivalent to a minimal encoder it becomes natural to consider only such encoders.

Note also that any minimal encoder is noncatastrophic [8], since any minimal encoder is basic and thus has a feedback-free inverse. Therefore, by considering only minimal encoders we automatically exclude catastrophic encoders, which are in general not desirable because they can cause infinite error propagation.

Let $w\{\boldsymbol{v}\}$ denote the Hamming weight of $\boldsymbol{v}$, and let $w_G(i)$ and $w_H(i)$ denote the weights of the ith row in $\boldsymbol{G}$ and the ith column in $\boldsymbol{H}$, respectively. To define the free distance, we now

use the fact that for each encoded sequence y the relation

$$yH^T = 0$$

must be satisfied.

Definition 3: The free distance is

$$d_{\text{free}} = \min_{Y} w\{y\}$$

where $Y = \{y: y \neq 0 \wedge yH^T = 0\}$.

In other words d_{free} is the smallest number of columns that add to zero in H.

Two more definitions are needed.

Definition 4: The reverse generator polynomial $G_i^{*j}(D)$ corresponding to $G_i^j(D)$ is $D^{v_i}G_i^j(D^{-1})$, and the reverse parity polynomial $H^{*j}(D)$ corresponding to $H^j(D)$ is $D^vH^j(D^{-1})$.

Definition 5: Let n be the number of the rightmost nonzero column among the first p rows in G, where $p \geq N - 1$. Then the (n,p)-terminated code corresponding to G is the (n,p) block code whose generator matrix is the leftmost n columns of the first p rows in G.

We shall mention here that the search for good convolutional codes is actually a search for minimal encoders generating codes with large values of d_{free}. However, to be in keeping with standard usage, throughout the remainder of this correspondence some of the properties precisely defined for encoders will be referred to as properties of the code generated by a given encoder.

III. Methods to Limit the Search for Good $(N,N-1)$ Convolutional Codes

Since an exhaustive search becomes practically impossible even for rather small values of v, some methods are needed to limit the search for *good* encoders, i.e., encoders generating codes with large values of d_{free}. Four different methods have been used here, plus a fifth method that was used partly to limit the search and partly to start the search in a subset in which the possibility of finding good encoders was "expected to be good." The methods were chosen because they result in simple rules that reject a large fraction of encoders from the complete ensemble. We shall mention also that some rules reject encoders because the free distance of the codes generated equals the d_{free} of some code in the remaining set, while others reject encoders because they cannot be good encoders. Further, it should be noted that we are interested only in minimal encoders and therefore rules that reject catastrophic encoders are included in the five methods no matter what the value of d_{free} might be for the codes generated by the rejected, catastrophic encoders.

The first method is to consider how the weights of the generators affect the achievable minimum distances d_p of the (n,p)-terminated codes.[1] To realize this we note first that the average weight of the code words in an (n,p)-terminated code is

$$D_p = \frac{n2^{p-1}}{2^p - 1}$$

where we have assumed the truncated generator matrix to be without all-zero columns.

Now let d be the desired free distance. Then every (n,p)-terminated code corresponding to G must have minimum distance $d_p \geq d$. This observation was used by Heller [7] to calculate the following upper bound

$$d_{\text{free}} \leq [\min_p D_p]$$

where $[x]$ denotes the largest integer not exceeding x. In almost all cases p is unique in this bound, but to be precise we define q to be the maximum value of p that minimizes D_p, and we call the corresponding (n,q)-terminated code the *critical terminated code* C_q.

Based on the generator weights $w_G(i)$ and d, we can also calculate a lower bound W_q on the average weight of the code words in C_q, but to get a reasonably strong bound three cases must be considered:

case 1: all q generators have even weights, d even;
case 2: s generators, $1 \leq s \leq q$, have odd weights, d even;
case 3: s generators, $1 \leq s \leq q$, have odd weights, d odd.

The derivation is as follows. There are $2^q - 1$ nonzero code words, q of which are equal to the generator rows. Since for all code words we must require the weight to be at least d, the total weight of the code words is lowerbounded by

$$W_T = \sum_{i=1}^{q} w_G(i) + (2^q - 1 - q)d.$$

This bound can be improved in the case where at least one of the generator rows has odd weight, since then exactly half the code words have odd weights. Accordingly, W_T is increased by α, the number of code words which are not already counted as generator rows and for which we can require the weight to be at least $d + 1$. The final expression then becomes

$$W = \frac{\sum_{i=1}^{q} w_G(i) + (2^q - 1 - q)d + \alpha}{2^q - 1}$$

where

$$\alpha = \begin{cases} 0, & \text{in case 1;} \\ 2^{q-1} - s, & \text{in case 2;} \\ 2^{q-1} - 1 - (q - s), & \text{in case 3.} \end{cases}$$

The following theorem is now obvious.

Theorem 1: If $W_q > D_q$, no C_q can exist with generator weights $w_G(i)$ and $d_q \geq d$, and therefore, also no corresponding convolutional code can exist with $d_{\text{free}} \geq d$.

Therefore, the first step is to calculate the weight possibilities for the code generators and then limit the search to the set of codes for which the code generators satisfy the calculated possibilities. Further, for each code in this set we check if $d_q \geq d$ and exclude the codes that fail this check.[2]

An example might be helpful. Say we want $N = 3$, $v = 9$, and $d = 10$. Then C_q is the (18,3) code, where the total weight of the code words is 72. Therefore, six code words have weight 10 and one code word has weight 12. Now assume $v_1 = 4$. Then only two weight possibilities exist:

a) $w_G(1) = 10 \quad w_G(2) = 10 \quad w\{g_1 + g_3\} = 10$ or 12,

b) $w_G(1) = 10 \quad w_G(2) = 12 \quad w\{g_1 + g_3\} = 10.$

(The third row of G equals the first row shifted three times to the right, and therefore it is very easy to check $w\{g_1 + g_3\}$ as soon as the first row is generated.)

The second method to reduce the search relies on the following theorem.

[1] This idea was originally used by Larsen [6].

[2] In case $d_q \geq d$ we shall say that the convolutional code has passed the critical code test.

Theorem 2: Consider the class of $(N, N-1)$ minimal encoders with overall constraint length ν. Then for any encoder $\boldsymbol{G}$ some encoder $\boldsymbol{G}'$ exists such that the following properties hold.

1) The free distance of the code generated by $\boldsymbol{G}'$ is equal to d_{free} of the code generated by $\boldsymbol{G}$.

2) $\boldsymbol{G}_0'$ has the form $[\boldsymbol{R};\boldsymbol{s}]$ where $\boldsymbol{R}$ is an $(N-1) \times (N-1)$ matrix with all ones on the diagonal and zeroes below.

3) All the rows of $\boldsymbol{G}_0'$ have even weight.

To prove this theorem we shall make use of three lemmas.

Lemma 1: Let C and C' be defined by the parity matrices

$$\boldsymbol{H}(D) = [H^1(D), H^2(D), \cdots, H^N(D)]$$

and

$$\boldsymbol{H}'(D) = [D^{-s_1}H^1(D), D^{-s_2}H^2(D), \cdots, D^{-s_N}H^N(D)]$$

respectively, where s_i is the largest number such that D^{s_i} divides $H^i(D)$. Then C and C' have equal free distances.

Proof: Assume C and C' have free distances d and d', respectively. Let $\boldsymbol{y}(D)$ represent any sequence of weight d such that

$$\boldsymbol{y}(D)\boldsymbol{H}^T(D) = \boldsymbol{0}.$$

Then $\boldsymbol{y}'(D) = [D^{s_1}y^1(D), D^{s_2}y^2(D), \cdots, D^{s_N}y^N(D)]$ also has weight d and

$$\boldsymbol{y}'(D)\boldsymbol{H}'^T(D) = \boldsymbol{0}$$

which implies $d' \leq d$. The opposite inequality $d \leq d'$ follows in a similar way, and therefore $d' = d$.

Lemma 2: Let $\boldsymbol{H}$ be the parity matrix corresponding to an $(N, N-1)$ minimal encoder $\boldsymbol{G}$. Then $\boldsymbol{H}_0 = [11 \cdots 1]$, if and only if all rows in $\boldsymbol{G}_0$ have even weight.

Proof: Note that $\boldsymbol{H}_0$ can be interpreted as the parity matrix of the $(N, N-1)$ block code whose generator matrix is $\boldsymbol{G}_0$. Then the lemma follows from the fact that the null space of $\boldsymbol{H}_0 = [11 \cdots 1]$ is exactly all the even-weight N-tuples, and any other $(N-1)$-dimensional space of N-tuples contains vectors of odd weight.

Lemma 3: Consider the class of $(N, N-1)$ minimal encoders with overall constraint length ν. Then for any encoder $\boldsymbol{G}$ some encoder $\boldsymbol{G}'$ exists such that $\boldsymbol{G}_0'$ has the form

$$\boldsymbol{G}_0' = [\boldsymbol{R};\boldsymbol{s}]$$

and the codes generated by $\boldsymbol{G}$ and $\boldsymbol{G}'$ have equal free distances.

Proof: Note first that row operations on some basic encoder [2] $\boldsymbol{G}(D)$ result in an equivalent basic encoder, and column permutations of $\boldsymbol{G}(D)$ correspond only to permutations of the output sequences, or, in other words, row operations on $\boldsymbol{G}(D)$ do not change the parity matrix, and column permutations of $\boldsymbol{G}(D)$ only change the order of the parity polynomials. Therefore, such operations result in codes with equal free distances. To realize $\boldsymbol{G}'(D)$ from $\boldsymbol{G}(D)$ without increasing ν in the obvious realization, we first order the rows of $\boldsymbol{G}(D)$ from top to bottom in order of increasing ν_i. Then any row above the ith can be added to the ith without increasing ν. But such "row operations from top to bottom" combined with column permutations are just enough to realize a $\boldsymbol{G}'(D)$ such that $\boldsymbol{G}_0'$ has the form $[\boldsymbol{R};\boldsymbol{s}]$.

Proof of Theorem 2: Let $\boldsymbol{G}$ be a minimal encoder generating a code with free distance d. From Lemma 1 it now follows that some code exists with free distance d and parity matrix $\boldsymbol{H}''$ such that $\boldsymbol{H}_0'' = [11 \cdots 1]$. Denote the corresponding minimal encoder by $\boldsymbol{G}''$. Then use Lemma 3 to construct from $\boldsymbol{G}''$ a minimal encoder $\boldsymbol{G}'$ such that $\boldsymbol{G}_0' = [\boldsymbol{R};\boldsymbol{s}]$; the code generated by $\boldsymbol{G}'$ also has free distance d. Finally, Lemma 2 ensures that all rows of $\boldsymbol{G}_0'$ have even weight.

Theorem 2 reduces the number β of different ways to realize $\boldsymbol{G}_0$ from

$$\beta = 2^{N(N-1)} \prod_{i=2}^{N} (2^i - 1)/2^i > 0.5 \times 2^{N(N-1)}$$

to $\beta = 2^{(N-1)(N-2)/2}$. However, β can be reduced further if we take advantage of the fact that we do not change the free distance of the codes as long as we use only row operations and column permutations. To ensure minimality of the encoder, the only restriction is that the row operations must not increase the constraint length of any row. Thus, for $N = 3$, it turns out that we need only to consider minimal encoders such that

$$\boldsymbol{G}_0 = \begin{pmatrix} 101 \\ 011 \end{pmatrix}.$$

If $N = 4$ and $\nu_1 < \nu_2 < \nu_3$, it is enough to consider three forms of $\boldsymbol{G}_0$, namely

$$\boldsymbol{G}_0 = \begin{pmatrix} 1001 \\ 0101 \\ 0011 \end{pmatrix} \quad \boldsymbol{G}_0 = \begin{pmatrix} 1111 \\ 0101 \\ 0011 \end{pmatrix} \quad \boldsymbol{G}_0 = \begin{pmatrix} 1010 \\ 0101 \\ 0011 \end{pmatrix}$$

while for $\nu_1 = \nu_2 < \nu_3$ or $\nu_1 < \nu_2 = \nu_3$ only the first two forms are needed, and for $\nu_1 = \nu_2 = \nu_3$ the first form suffices.

The third method we shall mention is based on a well-known property [5], which for the sake of completeness is stated as follows.

Theorem 3: Let $\boldsymbol{G}^*(D)$ denote the reverse encoder corresponding to $\boldsymbol{G}(D)$, i.e., the encoder with all generator polynomials reversed. Then the codes generated by $\boldsymbol{G}^*(D)$ and $\boldsymbol{G}(D)$ have equal free distances.

As a matter of form we shall mention here that if $\boldsymbol{H}(D)$ corresponds to $\boldsymbol{G}(D)$, then $\boldsymbol{H}^*(D)$, the parity matrix with all polynomials reversed, corresponds to $\boldsymbol{G}^*(D)$.

As the basis of the fourth method to limit the search we use the following theorem.

Theorem 4: Let $\boldsymbol{G}(D)$ generate a code with free distance d_{free}, let the corresponding parity matrix be $\boldsymbol{H}(D)$, and assume, without loss of generality, that

$$w_H(i) \leq w_H(j), \qquad \text{if } i < j.$$

Then the following properties hold:

1) $d_{\text{free}} \leq w_H(1) + w_H(2)$;
2) $d_{\text{free}} \leq 2(\nu + 1) + 2w_H(1) - w_H(N-1) - w_H(N)$;
3) If $w_H(i)$, $i = 1,2,\cdots,N$ are all odd, then d_{free} is even;
4) If $w_H(i)$, $i = 1,2,\cdots,N$ are all even, then $\boldsymbol{G}(D)$ is catastrophic.

Proof of Property 1: Let

$$\boldsymbol{y}(D) = [H^2(D), H^1(D), 0, 0, \cdots, 0].$$

Then $w\{\boldsymbol{y}(D)\} = w_H(1) + w_H(2)$ and $\boldsymbol{y}(D)\boldsymbol{H}^T(D) = H^2(D)H^1(D) + H^1(D)H^2(D) = 0$.

Proof of Property 2: Let

$$\boldsymbol{y}(D) = [H^{N-1}(D) + H^N(D), 0, 0, \cdots, 0, H^1(D), H^1(D)].$$

Then $w\{\boldsymbol{y}(D)\} = 2w_H(1) + w\{H^{N-1}(D) + H^N(D)\}$ and

$\boldsymbol{y}(D)\boldsymbol{H}^T(D) = 0$. Now, since we always have

$$w\{H^{N-1}(D) + H^N(D)\} \leq 2(v+1) - w_H(N-1) - w_H(N)$$

the property follows.

Proof of Property 3: Every sum of an odd number of columns with odd weights has odd weight. Therefore, an odd number of columns in $\boldsymbol{H}$ cannot add to the all-zero column (which has weight 0), and this implies that all codewords have even weight and hence that d_{free} is even.

Proof of Property 4: Note first that the determinants of the N distinct $(N-1) \times (N-1)$ submatrices of $\boldsymbol{G}(D)$ are exactly the parity polynomials of $\boldsymbol{H}(D)$. Then it follows from [8] that the encoder is noncatastrophic, if and only if for some L

$$\text{GCD}\{H^i(D),\ i = 1,2,\cdots,N\} = D^L.$$

But if $w_H(i)$ is even, then $(1+D)$ divides $H^i(D)$, which implies

$$\text{GCD}\{H^i(D),\ i = 1,2,\cdots,N\} \neq D^L.$$

With reference to Property 4, it is worth noting further that $\boldsymbol{G}(D)$ is catastrophic if a) all generator polynomials in a row of $\boldsymbol{G}(D)$ have even weight or b) in each $(N-1) \times (N-1)$ submatrix of $\boldsymbol{G}(D)$ one of the following situations occurs: at least one row contains even-weight polynomials only, at least one column contains even-weight polynomials only, or all $(N-1)^2$ polynomials have odd weight.

The fifth method we shall consider operates on $\boldsymbol{H}_v$.

Theorem 5: Let $\boldsymbol{H}$ represent a code with free distance d_{free}, and let the (n,q)-terminated code be the critical code C_q. If $\boldsymbol{H}_v$ contains s zeros, $0 < s < N$, then there exists an $(n+s, q+s)$ block code with $d_{\min} \geq d_{\text{free}}$.

Proof: Assume, without loss of generality, that $\boldsymbol{H}_v$ contains the s zeros as the first elements. Then notice that the parity matrix corresponding to the (n,q)-terminated code is the $(n-q) \times n$ matrix formed by the upper left corner of $\boldsymbol{H}$. Consider now instead the $(n-q) \times (n+s)$ matrix $\boldsymbol{H}'$ formed by the upper left corner of $\boldsymbol{H}$. Since d_{free} is the smallest number of columns that add to zero, $\boldsymbol{H}'$ must be a parity matrix for an $(n+s, q+s)$ block code with $d_{\min} \geq d_{\text{free}}$.

In general, the requirement for an $(n+s, q+s)$ code with $d_{\min} \geq d$ is stronger than the requirement for an (n,q) code with $d_{\min} \geq d$. Therefore, in most cases we need only to search among codes with $\boldsymbol{H}_v = [11\cdots1]$ to maximize d_{free}. However, there are also cases where $(n+s, q+s)$ block codes exist with $d_{\min} \geq d$. But in such cases we expect the possibilities of finding codes with $d_{\text{free}} = d$ to be better if we search in the subset where $\boldsymbol{H}_v = [11\cdots1]$. This is what is meant by a subset where the possibilities are "expected to be good."

IV. Search Procedure for Optimal Codes

Theorems 1–5 form the basis of a search procedure that was used to search for optimal (in the sense of maximizing d_{free}) rate-2/3 and 3/4 codes, but for simplicity the search procedure shall be stated only for rate-2/3 codes. For each value of v the procedure is as follows.

Step 1: Find the best upper bound $D_q(v)$ that can be evaluated by the method given by Heller [7],[3] and determine the parameters of C_q. Set the desired free distance d equal to $D_q(v)$.

Step 2: Use the parameters of C_q and d to determine corresponding weight possibilities $w_G(1)$, $w_G(2)$, and eventually $w\{\boldsymbol{g}_1 + \boldsymbol{g}_3\}$ and $w\{\boldsymbol{g}_2 + \boldsymbol{g}_4\}$. This is easily done by hand calculation. Then define ω to be the set of remaining encoders that comply with the following conditions.

a) The weight of the generators corresponds to the possibilities just determined.

b) $\boldsymbol{G}_0 = \begin{pmatrix}101\\011\end{pmatrix}$.

c) $\boldsymbol{H}_v = [111]$.

d) No generator contains even-weight polynomials only.

By remaining encoders, we mean that once an encoder is rejected by the search procedure it is deleted from ω.

Step 3: Pick at random an encoder $\boldsymbol{G}$ from ω, and check whether the code passes C_q. Note that $\boldsymbol{G}$ can be rejected once a code word of weight less than d is found. In case $\boldsymbol{G}$ is rejected, go to step 6.

Step 4: Calculate the parity matrix, and check whether $\boldsymbol{G}$ can be rejected according to theorem 4. If so, go to step 6.

Step 5: Calculate d_{free}. In case $d_{\text{free}} = d$, terminate the procedure.

Step 6: Delete $\boldsymbol{G}$ from ω; if the reverse encoder is in ω, delete this encoder also. In case ω is empty, check whether codes with $\boldsymbol{H}_v \neq [111]$ and $d_{\text{free}} = d$ can exist in view of Theorem 5. If so, redefine ω by skipping condition c) and return to Step 3; otherwise, decrease d by one and return to Step 2.

A few remarks on the calculation of d_{free} are in order. The method used is the bidirectional search algorithm originally presented by Bahl *et al.* [9], but corrected by Larsen [10]. However, instead of using the states in the encoder, we use the states in the syndrome-former, which is of course possible since a linear correspondence exists between the states of the two [1]. One reason for using the syndrome-former instead of the encoder is that we can gain in speed. We shall explain this a bit further by referring to the algorithm in [10]. In Step 4 and Step 5 in [10], we compute the weights of the extensions. This parity calculation is, on most general-purpose computers, rather time-consuming. Now consider the syndrome-former S for a rate-$(N-1)/N$ code. If the output $\boldsymbol{s}$ from S is the all-zero sequence and S is driven by rational sequences, then those sequences are codewords. But the output at time u, s_u, depends only on the content α_u of the last memory element in S and the input at time u, y_u. Therefore, knowing $\boldsymbol{H}$, it becomes easy in an initialization step to split the extension possibilities into two sets $Y(\alpha)$, $\alpha = 0,1$. In each set we get 2^{N-1} values of y_u such that $s_u = 0$. Now we realize that all the extensions at step u are in $Y(\alpha_u)$, but since we use the elements of $Y(\alpha_u)$ directly to drive S, we also know the weight of each extension, and thus we need compute neither the extension nor the weight of the extension. Finally, we shall mention that the next state of S is easily determined.

In Table I we have listed the upper bounds $D_q(v)$, and in Tables II–V the codes found by the search procedure. They are all optimum in the sense that no code exists with equal rate and constraint length but a larger value of d_{free}, and no code exists with equal rate and free distance, but a smaller value of v. In the cases where the bounds in Table I are achieved, the optimality is obvious, but in the other cases very little search was enough to show that $d_{\text{free}} = D_q(v)$ was unachievable. Two examples for the rate-2/3 codes follow.

Example 1: $d_{\text{free}} = D_q(6) = 8$ is unachievable. The weight calculations in Steps 1 and 2 show that such a code would require the (12,2) and (15,4) terminated codes to be equidistant

[3] In fact, any upper bound on (n,p)-terminated block codes may be used. In the actual case the bound in [11] was used.

TABLE I
UPPER BOUNDS FOR SHORT RATES 2/3 AND 3/4 CONVOLUTIONAL CODES

constraint length	free distance			
	R=2/3		R=3/4	
ν	upper bound	achieved	upper bound	achieved
2	4	3		
3	4	4	4	4
4	6	5	4	(4)
5	6	6	6	5
6	8	7	6	6
7	8	8	8	(6)
8	8	(8)	8	7
9	10	9	8	8
10	10	10	9	

TABLE II
PARITY POLYNOMIALS OF SOME OPTIMAL (3,2) CONVOLUTIONAL CODES (OCTAL FORM)

ν	$H^1(D)$	$H^2(D)$	$H^3(D)$	d_{free}
2	4	5	7	3
3	13	15	11	4
4	23	35	31	5
5	51	53	65	6
6	163	145	105	7
7	367	271	301	8
9	1255	1121	1527	9
10	3543	3177	2415	10

TABLE III
PARITY POLYNOMIALS OF SOME OPTIMAL (4,3) CONVOLUTIONAL CODES (OCTAL FORM)

ν	$H^1(D)$	$H^2(D)$	$H^3(D)$	$H^4(D)$	d_{free}
3	11	17	13	15	4
5	51	47	63	45	5
6	113	105	177	111	6
8	657	575	727	431	7
9	1243	1725	1565	1071	8

codes. Thus $w_G(1) = w_G(2) = w\{g_1 + g_3\} = w\{g_2 + g_4\} = 8$. Further, $H_6 = [111]$ should hold. It turns out that there are no more than 18 different ways to generate g_1, and a total of only 167 encoders were generated in Step 3. Among those, 100 encoders were rejected immediately because $w\{g_1 + g_2\} \neq 8$, and a total of 148 encoders were rejected in Step 3, while the remaining 19 encoders were rejected in Step 4.

Example 2: $d_{free} = D_q(9) = 10$ is unachievable. To show this by the search procedure, d_{free} was computed for fewer than 20000 codes; the entire check took about 1 min of central processing unit (CPU) time on an IBM 370/165.

V. CONCLUSION

The codes found by the search procedure are short but optimum codes, and it is noteworthy that all the rules used to limit the search are simple rules. In view of that, it may appear surprising that up to now little progress has been made in finding good high-rate convolutional codes. However, Forney's results [1] will hopefully stimulate interest in this direction.

TABLE IV
GENERATOR MATRICES OF THE (3,2) CONVOLUTIONAL CODES IN TABLE II

ν	$\underline{G}_0$	$\underline{G}_1$	$\underline{G}_2$	$\underline{G}_3$	$\underline{G}_4$	$\underline{G}_5$
2	101 011	111 100				
3	101 011	011 001	000 101			
4	101 011	100 101	110 011			
5	101 011	111 001	011 101	000 101		
6	101 011	111 111	010 101	101 011		
7	101 011	110 001	011 101	011 111	000 110	
9	101 011	001 010	101 011	011 100	110 001	000 101
10	101 011	100 111	010 100	011 010	101 100	110 011

TABLE V
GENERATOR MATRICES OF THE (4,3) CONVOLUTIONAL CODES IN TABLE III

ν	$\underline{G}_0$	$\underline{G}_1$	$\underline{G}_2$	$\underline{G}_3$
3	1111 0101 0011	0000 0110 0100	0000 0000 0011	
5	1001 0101 0011	1111 0101 0100	0000 1001 0011	
6	1001 0101 0011	1001 1001 1110	0101 1010 0110	
8	1001 0101 0011	1110 0000 0010	1100 1101 0110	0000 1001 1010
9	1001 0101 0011	0011 0111 1011	0110 0001 1000	0110 1100 1001

ACKNOWLEDGMENT

The author wishes to thank Prof. J. L. Massey, University of Notre Dame, Notre Dame, Ind., together with K. J. Larsen and J. Justesen, both at the Laboratory for Communication Theory, Technical University of Denmark, for some helpful suggestions. Further, the reviewers are acknowledged for their valuable suggestions which helped to clarify the final manuscript.

REFERENCES

[1] G. D. Forney, Jr., "Structural analysis of convolutional codes via dual codes," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 512–518, July 1973.
[2] ——, "Convolutional codes I: Algebraic structure," *IEEE Trans. Inform. Theory*, vol. IT-16, pp. 720–738, Nov. 1970.
[3] ——, "Convolutional codes II: Maximum likelihood decoding," Stanford Univ., Stanford, Calif., Tech. Rep. 7004-1, June 1972; also *Inform. Contr.*, 1974, to be published.
[4] D. J. Costello, "Construction of convolutional codes for sequential decoding," Dep. Elec. Eng., Univ. Notre Dame, Tech. Rep. EE-692, Aug. 1969.
[5] Viterbi *et al.*, "Concatenation of convolutional and block codes," Univ. Calif., Los Angeles, Final Rep., N71-32505, June 1971.
[6] K. J. Larsen, "Short convolutional codes with maximal free distance for rates 1/2, 1/3, and 1/4," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 371–372, May 1973.

[7] J. A. Heller, "Sequential decoding: Short constraint length convolutional codes," in *Jet Propul. Lab., Calif. Inst. Tech., Pasadena, Space Programs Summary* 37-54, vol. 3, pp. 171–174, Dec. 1968.
[8] J. L. Massey and M. K. Sain, "Inverses of linear sequential circuits," *IEEE Trans. Comput.*, vol. C-17, pp. 330–337, Apr. 1968.
[9] L. R. Bahl, C. D. Cullum, W. D. Frazer, and F. Jelinek, "An efficient algorithm for computing free distance," *IEEE Trans. Inform. Theory* (Corresp.), vol. IT-18, pp. 437–439, May 1972.
[10] K. J. Larsen, "Comments on 'An efficient algorithm for computing free distance,'" *IEEE Trans. Inform. Theory* (Corresp.), vol. IT-19, pp. 577–579, July 1973.
[11] H. J. Helgert and R. D. Stinaff, "Minimum-distance bounds for binary linear codes," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 344–356, May 1973.

Uniform Complex Codes with Low Autocorrelation Sidelobes

U. SOMAINI AND MARTIN H. ACKROYD

***Abstract*—The problem of designing uniform-amplitude codes with good autocorrelation functions can be regarded as a problem of minimizing a function of several continuous variables. The application of numerical methods of minimization is shown to yield codes with lower sidelobe levels than other known codes of equal lengths. Codes with no sidelobe exceeding unity have been found for lengths as large as 18.**

Finite-length complex number sequences or codes having good autocorrelation functions are of interest in radar and communication system design. The autocorrelation function of an $(N+1)$-element sequence $\{c_k\}$ is given by

$$r_n = \sum_{k=0}^{N-n} c_k c_{k+n}^*, \qquad n = 0, \pm 1, \cdots, \pm N.$$

Huffman [1] showed how complex sequences of lengths $N+1$ could be found having near ideal autocorrelation function; i.e., the autocorrelation sequence is zero, for all shifts n at which this is theoretically possible. However, all known Huffman codes with more than three elements are nonuniform; that is, their elements are not all of the same magnitude. In comparison to uniform codes, this is a severe disadvantage because a nonuniform code requires an amplitude modulator in its implementation and has less energy for a given length. For this reason, uniform (constant-amplitude) sequences of complex numbers with low autocorrelation sidelobes are of particular interest.

The most extensively studied uniform sequences are the binary codes. However, it does not yet seem possible to find long binary codes that have very low sidelobe levels. Even the best of the known binary codes [2] have autocorrelation sidelobe energies, $\sum_{n=1}^{N} r_n^2$, which are about one tenth of the mainlobe energy, r_0^2.

Polyphase codes are uniform codes whose elements have phases that are integer multiples of a basic phase angle $2\pi/M$. Some polyphase codes, such as the Frank codes [3], have lower sidelobe levels than the best available binary codes of the same length. Golomb and Scholtz [4] speculate that polyphase codes of all lengths exist that have no sidelobe with a magnitude exceeding unity and whose elements have just six phases. So far no sextic codes of lengths greater than 15 have been found [5]. [4] gives a four-phase code of length 15 attributed to Carley. Develet [6] has shown that Carley's code can be extended to an eight-phase code of length 16.

Manuscript received October 24, 1973; revised March 15, 1974.
The authors are with the Department of Electronic and Electrical Engineering, University of Technology, Loughborough, Leicestershire, England.

TABLE I

Length of the sequence	Peak sidelobe magnitude		Sidelobe energy as percent of mainlobe energy	
N+1	$\max_n \lvert r_n \rvert$		$100 \sum_{n=1}^{N} \lvert r_n \rvert^2 / r_0^2$	
	prior	new	prior	new
4	(B) 1.00	1.00	12.50	9.38
5	(B) 1.00	1.00	8.00	8.01
6	(G) 1.00	1.00	13.89	13.89
7	(B) 1.00	1.00	6.13	3.32
8	(G) 1.00	1.00	7.81	4.32
9	(G) 1.00	1.00	2.41	1.32
10	(G) 1.00	1.00	8.64	4.70
11	(B) 1.00	1.00	4.14	2.69
12	(G) 1.00	1.00	6.25	4.71
13	(B) 1.00	1.00	3.55	2.87
14	(S) 1.00	1.00	5.61	3.76
15	(C) 1.00	1.00	3.11	2.38
16	(D) 1.00	1.00	4.73	3.17
17		1.00		3.32
18		1.00		3.16
19		1.08		3.34
20		1.14		3.12
21		1.28		3.12
23		1.20		2.61
25	(F) 1.62	1.17	4.42	2.22
27		1.39		3.49
29		1.50		3.80
31		1.31		3.27
33		1.69		4.32
35		1.66		3.82
36	(F) 2.00	1.46	3.86	2.66
37		1.58		3.46
39		1.84		4.33
41		1.93		4.63
43		1.83		3.28
49	(F) 2.25	1.49	3.10	1.62
64	(F) 2.61	1.67	2.80	1.43
81	(F) 2.88	1.89	2.37	1.59
100	(F) 3.24	1.86	2.20	1.44

Autocorrelation sidelobe levels of codes obtained by numerical optimization together with those of Barker, Golomb and Scholtz, Scholtz, Carley, Develet and Frank, indicated by B, G, S, C, D, F, respectively.

All available methods for discovering new binary or polyphase codes seem to contain an element of trial and error. One approach is repeatedly to choose codes at random and to evaluate their autocorrelation functions until a satisfactory code is found. Another approach [2], [7] is to choose an initial code, perhaps at random, and from it to produce a succession of progressively better codes. This is done by taking the current code, changing the phases of one or more of its elements in some way and evaluating the autocorrelation function of the resulting code. If some measure of the sidelobes is reduced, the modification is retained and the new code is subjected to further modification. The problem is in effect one of minimizing a function of a number of discrete-valued variables.

Instead of searching for good polyphase codes, one can instead search for uniform complex codes whose phases can have any values. This has the advantage that numerical methods for minimizing functions of continuous-valued variables can be used, and it must give at least as good results, since uniform complex codes include polyphase codes as a special case. The initial results given here suggest that the method of numerical optimization can yield useful codes within reasonable economy of computing effort.

A measure of the autocorrelation sidelobes such as that given by

$$I = \sum_{n=1}^{N} |r_n|^p$$